Lezione n.10
Laura Ricci

Freenet
Materiale didattico:
articoli distribuiti a lezione
Laura Ricci

IL PROGETTO FREENET

- Progetto Freenet: prende spunto da un progetto sviluppato nel 1999 da Ian Clarke, uno studente undergraduate (laurea triennale) di Edinburgo

- Contemporaneo di Napster, Gnutella. Svantaggio principale di questi sistemi: non garantiscono l'anonimato
  - È possibile individuare l'host da cui si sta scaricando materiale
  - È possibile individuare chi ha inviato una query

- Obiettivo fondamentale: garantire l'anonimato, rendere più ardua la censura del materiale condiviso

- Obiettivi secondari:
  - disponibilità dell'informazione
  - affidabiltà
  - efficienza
  - scalabilità

IL PROGETTO FREENET

- Luglio 1999: pubblicazione di un primo articolo contenente le idee fondamentali alla base del sistema
- Successivamente: Clarke ed un gruppo di volontari iniziano a lavorare al progetto. I membri del gruppo di lavoro appartengono a diversi paesi
- Sviluppo di Freenet avviene in modo decentralizzato
- Marzo 2000: versione 0.1 di Freenet
- 2000: il lavoro di Clarke "Freenet: A Distributed Anonymous Information Storage and Retrieval System" risulta il lavoro più citato nel campo della computer science, nel 2000 (fonte Citeseer)

FREENET VS: NAPSTER, GNUTELLA

- Napster, Gnutella: forniscono un servizio di condivisione di files.
  - i files sono pubblicati sulla memoria dell'host locale
  - i files rimangono disponibili sulla rete solo durante il periodo in cui l'utente rimane on-line

- Freenet: fornisce un servizio di memorizzazione e ricerca distribuita di files
  - ogni nodo mette a disposizione uno spazio di memorizzazione
  - l'informazione pubblicata viene memorizzata su un insieme di nodi della overlay network
  - l'informazione rimane sulla rete anche quando l'host che l'ha pubblicata si disconnette
  - rete Freenet = file system distribuito

FREENET:ROUTING

- Content Based Routing: Algoritmo di routing guidato dalla chiave che si vuole ricercare (come per le DHT)

- Algoritmo di routing
  - basato su euristiche
  - non garantisce l'assenza di falsi negativi

- Routing adattivo: il grafo delle connessioni logiche tra i nodi (overlay network) evolve nel tempo verso una stabilità ed efficienza maggiore,i nodi si specializzano nella gestione di clusters di dati

- La rete che Freenet che si crea mano a mano che nuovi nodi si inseriscono nella rete può essere considerata una rete small world

FREENET:CARATTERISTICHE GENERALI

- Resilienza: capacità di un sistema di adattarsi alle condizioni d'uso e di resistere all'usura in modo da garantire la disponibilità dei servizi erogati.Sinonimi: elasticità

- Resilienza della rete Freenet: Freenet può perdere una rilevante percentuale di nodi senza un'apprezzabile riduzione delle prestazioni e la maggioranza dei suoi nodi senza cessare di funzionare

- Comportamento "ecologico": l'informazione può essere inserita in Freenet, ma non rimossa: può essere solo lasciata "morire" di morte naturale. L'informazione che viene richiesta si moltiplica su più nodi e si "avvicina" ai nodi che la richiedono; quella non richiesta scomapre

FREENET: CARATTERISTICHE GENERALI

- L'utente non possiede controllo o conoscenza di quali files sono memorizzati nel proprio nodo
- Tutti i files all'interno del data store di un nodo sono crittografati
- Questa scelta viene fatta per proteggere l'utente dalla responsabilità sul contenuto dei file memorizzati nel proprio data store
- Comportamento non deterministico: il funzionamento di Freenet è non deterministico e non consente di provare con certezza che un certo file memorizzato nel datastore sia stato richiesto dal nodo locale e non da un altro nodo sulla rete

COSA FREENET NON PUO' FARE

- Non esiste la possibilità di indicizzare le chiavi in modo da effettuare query complesse
- Esistono alcune proposte (solo a livello teorico) per la creazione di indici interni a Freenet, ma nessuna proposta non è stata implementata
- Meccanismo di bootstrap 'primitivo':
  - per bootstrappare un nuovo nodo occorre conoscere l'indirizzo di almeno un "nodo affidabile" di Freenet
  - Si utilizza un server web e/o un file aggiornato di nodi distribuito insieme ai files di supporto di Freenet

FREENET: IMPLEMENTAZIONE

- I nodi comunicano tra di loro mediante il semplice protocollo connection-oriented FNP (Freenet Network Protocol) realizzato sopra TCP/IP
- I client applicativi che vogliono utilizzare i servizi di Freenet di un nodo locale utilizzano un altro protocollo chiamato FCP (Freenet Client Protocol)
- Implematati diversi clients (Frost, FMB Freenet Message Board, Freeweb, Manifest, FCPtools)
- Realizzato in JAVA
- Interfaccia utente permette di operare in maniera intuitiva, ma anche di controllare aspetti molto tecnici del protocollo

FREENET: LE CHIAVI

- In Freenet i files sono identificati mediante chiavi assegnate ad essi nel momento in cui il file viene inserito nella rete ed utilizzati per reperirli

- Esistono tre diversi tipi di chiavi
  - CHK – content hash key
  - SSK – signed subspace key
  - KSK – keyword signed key

- Ogni chiave Freenetpossiede la seguente struttura
  - "freenet:" è il prefisso standard
  - I primi tre caratteri indicano il tipo di chiave: SSK, KSK, CHK
  - Il simbolo"@" separa il tipo della chiave dal resto della chiave
  - Seguono un insieme di caratteri che identificano il file condiviso
  - Ad esempio: freenet: KSK@papers/p2p/freenet/keys

CHK: CONTENT HASHED KEY

- Content Hashed Key: viene generata calcolando un hash del contenuto del file
- Non esistono due file diversi con la stessa CHK
- Utilizzata come identificatore a basso livello del file GUID (o per file che non vengono modificati), poco user friendly
- Utilizzata insieme a meccanismi di più alto livello
- Generata mediante 160-bit SHA-1
- Il file viene crittato mediante una chiave generata in modo casuale
- La chiave per la decrittazione non viene mai memorizzata con il file, ma pubblicata insieme alla chiave CHK del file
- Utilizzata per riconoscere copie identiche dello stesso file, ma raramente per la ricerca

CHK: CONTENT HASHED KEY

- Esempio:freenet:CHK@SVbD9~HM5nzf3AX4yFCBc-A4dhNUF5DPJZLL5NX5Brs,bA7qLNJR7IXRKn6uS5PAySjIM6azPFvK~18kSi6bbNQ,AAEA--8
- La prima parte della chiave rappresenta il risultato dell'hash, la seconda parte è una chiave utilizzata per crittografare il file
- Il file viene crittografato non tanto per ragioni di sicurezza, poiché ciascuno deve essere in grado di decrittare il file, una volta reperito
- Le chiavi non vengono memorizzate insieme al file: un utente non conosce il contenuto dei file memorizzati nel proprio data store

GARANTIRE L'ANONIMATO

- Gli utenti non conoscono il contenuto dei propri data stores
- Chi inserisce un dato sulla rete deve crittarlo
- Le chiavi per crittografare i files
  - non vengono utilizzate nel routing e non vengono incluse nei messaggi scambiati dal protocollo
  - Vengono distribuite agli uetenti che possono decrittare i files dopo che gli hanno reperiti
- I nodi intermedi conoscono solo la chiave binaria del file (complesso invertire la funzione hash)

FREENET: SIGNED SUBSPACE KEY

- signed-subspace key (SSK): definiscono dei sottospazi di chiavi
- Sottospazio di chiavi: solo un utente può modificare i dati in quel sottospazio, tutti possono reperire i dati da quel sottospazio se conoscono la chiave di accesso
- Come vengono create le SSK
  - si crea in modo casule una coppia chiave pubblica/chiave privata
  - l'utente sceglie una breve descrizione per il file
  - si calcola separatamente un hash della parte pubblica della chiave generata e della descrizione fornita dall'utente
  - si concatenano le due stringhe ottenute dall'hash e si genera un nuovo hash, questo rappresenta la chiave binaria del file (GUID)
  - Si utilizza la parte privata della chiave per firmare il file

CHIAVI PUBBLICHE/PRIVATE

- Crittografia Asimmetrica: è una forma di crittografia che utilizza una coppia di chiavi: la chiave pubblica e la chiave privata
- La chiave privata viene mantenuta segreta, mentre quella pubblica viene pubblicata
- Le chiavi sono legate tra di loro da una relazione matematica, ma la complessità di derivare la chiave privata da quella pubblica è molt alta
- Utilizzate per:
  - Crittaggio/decrittaggio dei messaggi. Un messaggio (o un file) viene crittato utilizzando la chiave pubblica può essere decrittato utilizzando la chiave privata
  - Firmare i messaggi (o i dati) La firma di un messaggio può essere generata utilizzando la chiave privata. L'integrità di un messaggio firmato mediante la chiave privata può essere verificata da qualsiasi nodo che abbia accesso alla corrispondente chiave pubblica

FREENET: LE CHIAVI

- Il nodo che inserisce il file applica una funzione hash al contenuto del file o ad una sua parte
- La firma del file viene ottenuta crittando il risultato dell'hash mediante la chiave privata. La firma viene memorizzata insieme al file
- ogni client può verificarne l'integrità dei files che scaricato. Il client
  - applica la stessa funzione hash al contenuto del file (o di una sua parte)
  - decritta, mediante la chiave pubblica la firma associata al file
  - Se i due valori coincidono, il file è integro altrimenti è stato manomesso

FREENET: LE CHIAVI

Per reperire un file

- Si calcola la chiave binaria (GUID) del file sulla base della sua descrizione e del namespace (chiave pubblica) all'interno del quale si trova il file
- Viene generata una query contenente il GUID del file
- Problema: conoscere le chiavi pubbliche che identificano i diversi namespace
- (meccanismi di tipo keyring,...)

Per aggiornare un file

- occorre possedere la sua chiave privata
- il nodo che vuole aggiornare il file deve produrre una firma valida per esso
- l'aggiornamento dei files può avvenire solo da parte di chi li ha inseriti nel sistema

KSK: KEYWORLD SIGNED KEY

- Chiavi KSK (Keyworld Signed Key) sono un sottotipo delle SSK
  - La coppia chiave pubblica/chiave privata viene generata a partire da una dscrizione del file fornita dall'utente
  - La chiave associata al file viene costruita mediante hash della chiave pubblica
  - Il reperimento di un documento avviene a partire da questa descrizione
  - Soluzione più user friendly, ma meno sicura

FEENET:IL ROUTING

- ogni nodo Freenet fornisce uno spazio di memorizzazione
- il file viene inizialmente memorizzato su un insieme di nodi della rete
- un file può successivamente essere replicato su altri nodi
- Routing
  - Utilizza i GUID dei file per ricercarli
  - guidato dalla conoscenza della chiave del file da ricercare
  - Routing adattivo: le tabelle di routing vengono modificate durante la ricerca delle chiavi

FREENET: CARATTERISTICHE GENERALi

- Ogni nodo memorizza
  - un insieme di files
  - una tabella di routing che contiene
    - riferimenti ad altri nodi
    - alcune chiavi che individuano files memorizzati su tali nodi

- i riferimenti contenuti nelle routing tables definiscono la overlay network

- Routing:
  - basato su steepest descendant hill-climbing serch
  - associa HTL (Hops to Live) alle queries per limitarne la diffusione
    - L'HTL può essere incrementato, fino ad un valore massimo
  - associa identificatori casuali di 64 bits alle queries, per evitare loops
    - ogni nodo memorizza una lista degli identificatori delle queries già inoltrate

FREENET:ROUTING

Algoritmo di Routing: quando N riceve la chiave K

- se N possiede il file corrispondente a K, la query non viene inoltrata
  - N invia il file al nodo che lo aveva richiesto.
  - il file viene propagato indietro lungo il cammino P percorso dalla query per raggiungere N,

- se N non possiede il file
  - ricerca nella routing table la chiave K' numericamente più vicina a K
  - se K' esiste, inoltra K al nodo N' associato a K'
  - attende una risposta da N'

FREENET:ROUTING

Quando un nodo N riceve una risposta ad una query da un nodo vicino V

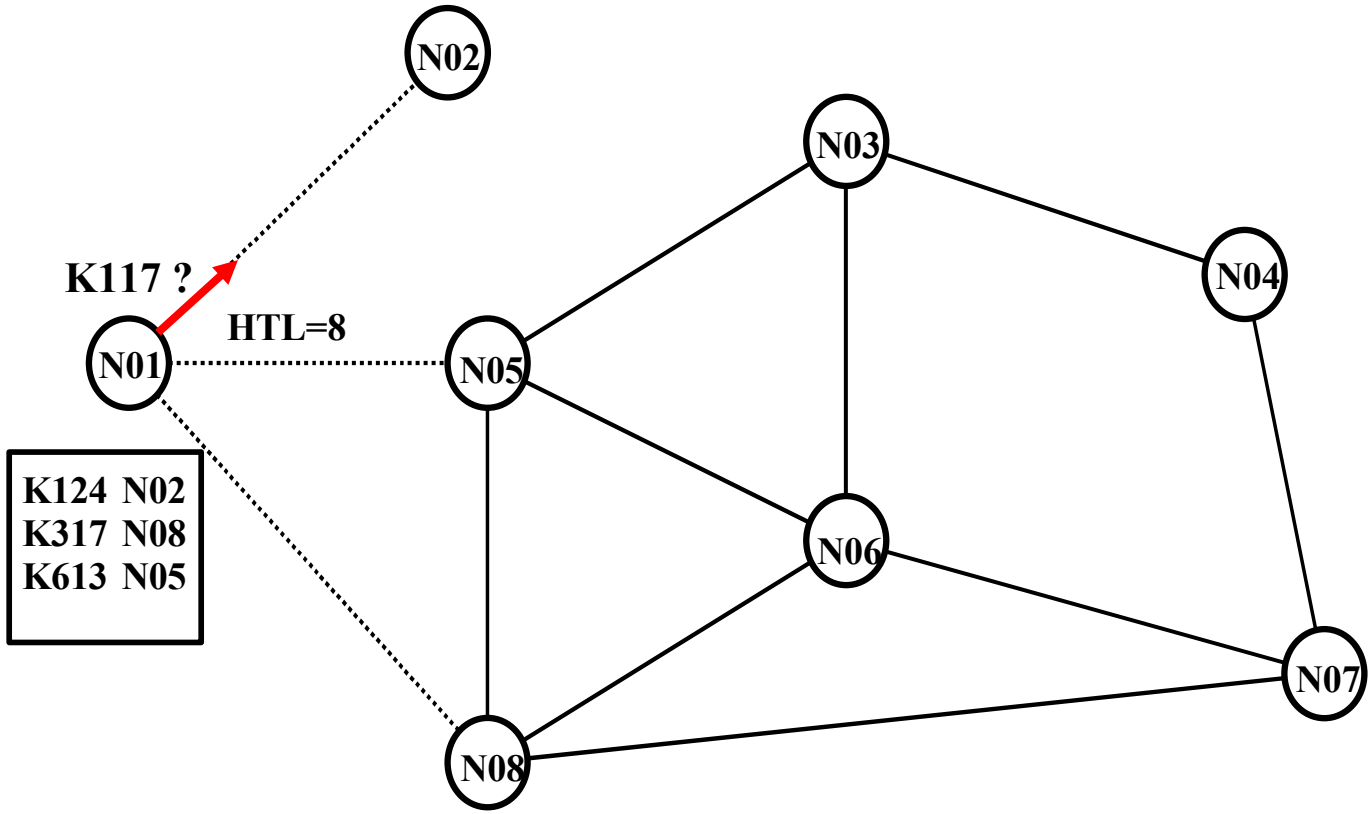
- se V restituisce un messaggio di fallimento, request failed, cerca la chiave numericamente più vicina a K, scelta tra le rimanenti
  - quando tutti i vicini sono stati contattati senza successo, restituisce un messaggio request failed al vicino che gli aveva inviato la query

- se V invia una risposta positiva (K,S), dove S identifica il nodo che possiede la chiave
  - Invia la risposta indietro al nodo da cui aveva ricevuto la query
  - aggiunge una nuova entrata (K,S) alla propria routing table
  - può memorizzare una copia del file nella memoria locale (dipende dalla sua distanza dal nodo S)

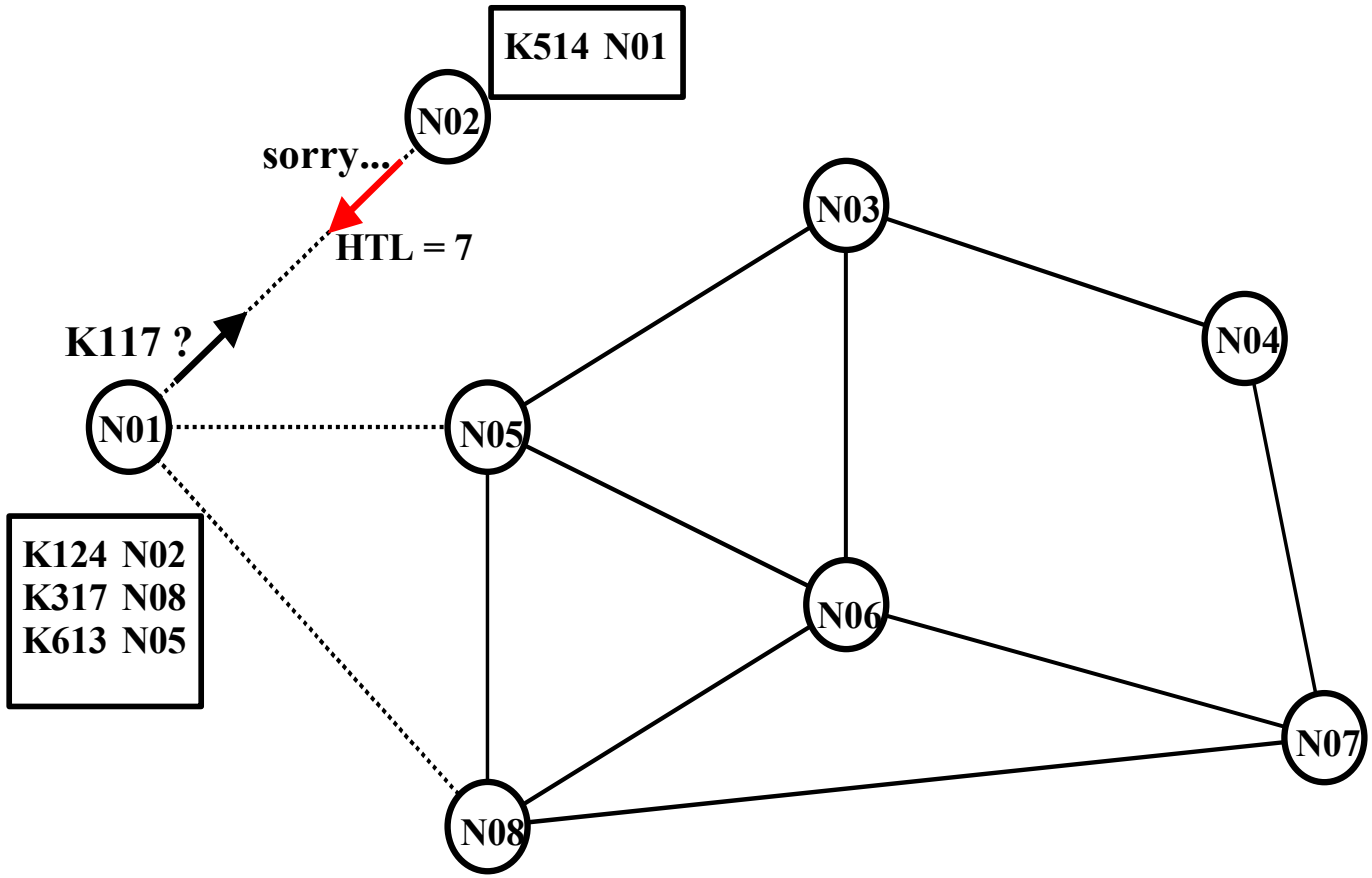
FREENET ROUTING

- N può rispondere ad una successiva query identica prelevando i dati dal proprio database
- Una richiesta successiva per una chiave "simile" viene inoltrata direttamente al nodo S
- La tabella di routing viene modificata dinamicamente mano a mano che le queries vengono risolte
- La tabella di routing di un nodo potrebbe essere utilizzata per reperire i nodi che hanno pubblicato i files
- Per aumentare l'anonimato, ogni nodo può decidere di cambiare nel messaggio di reply ad una query l'identità del nodo che fornisce il file

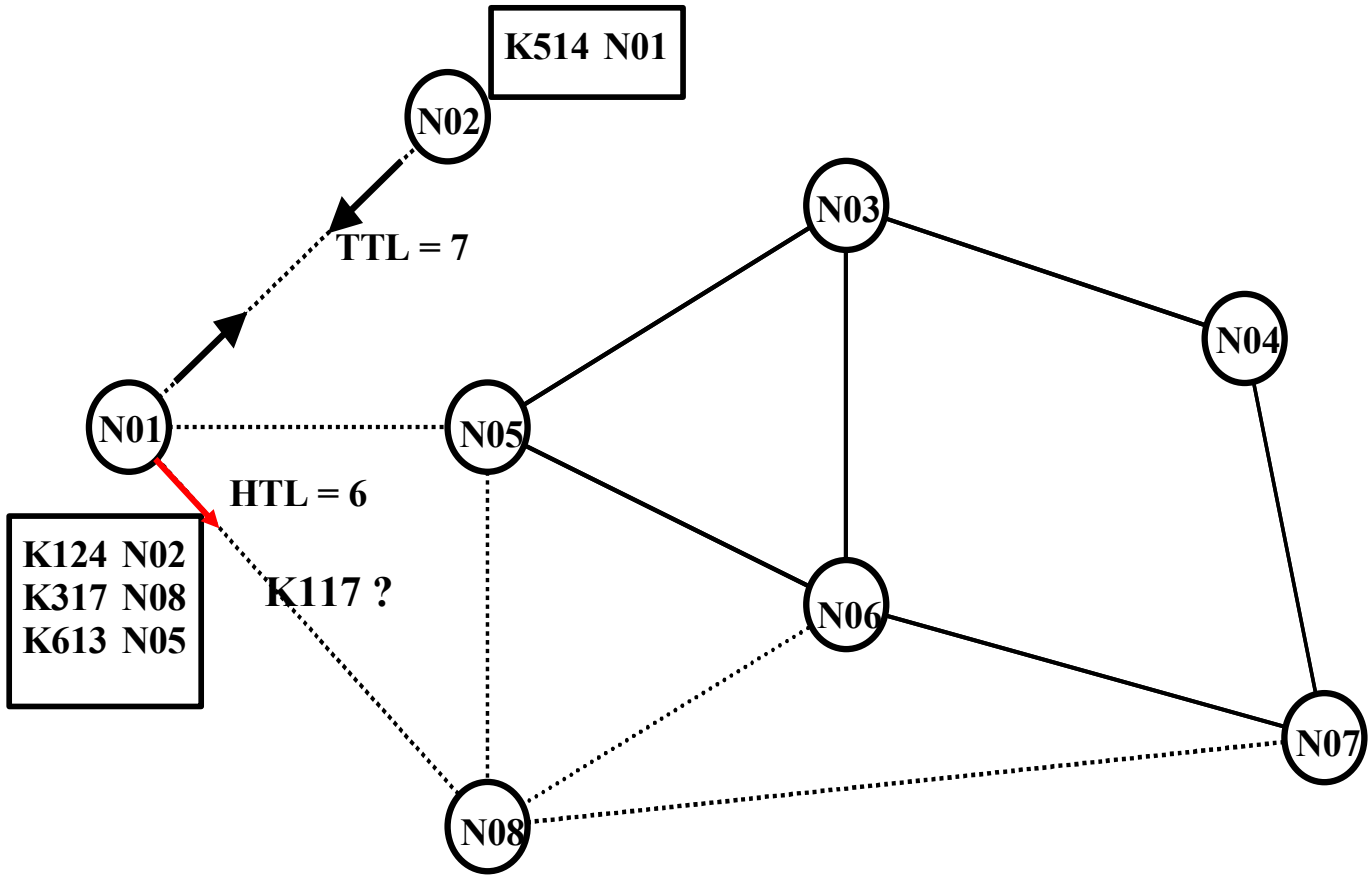
FREENET ROUTING



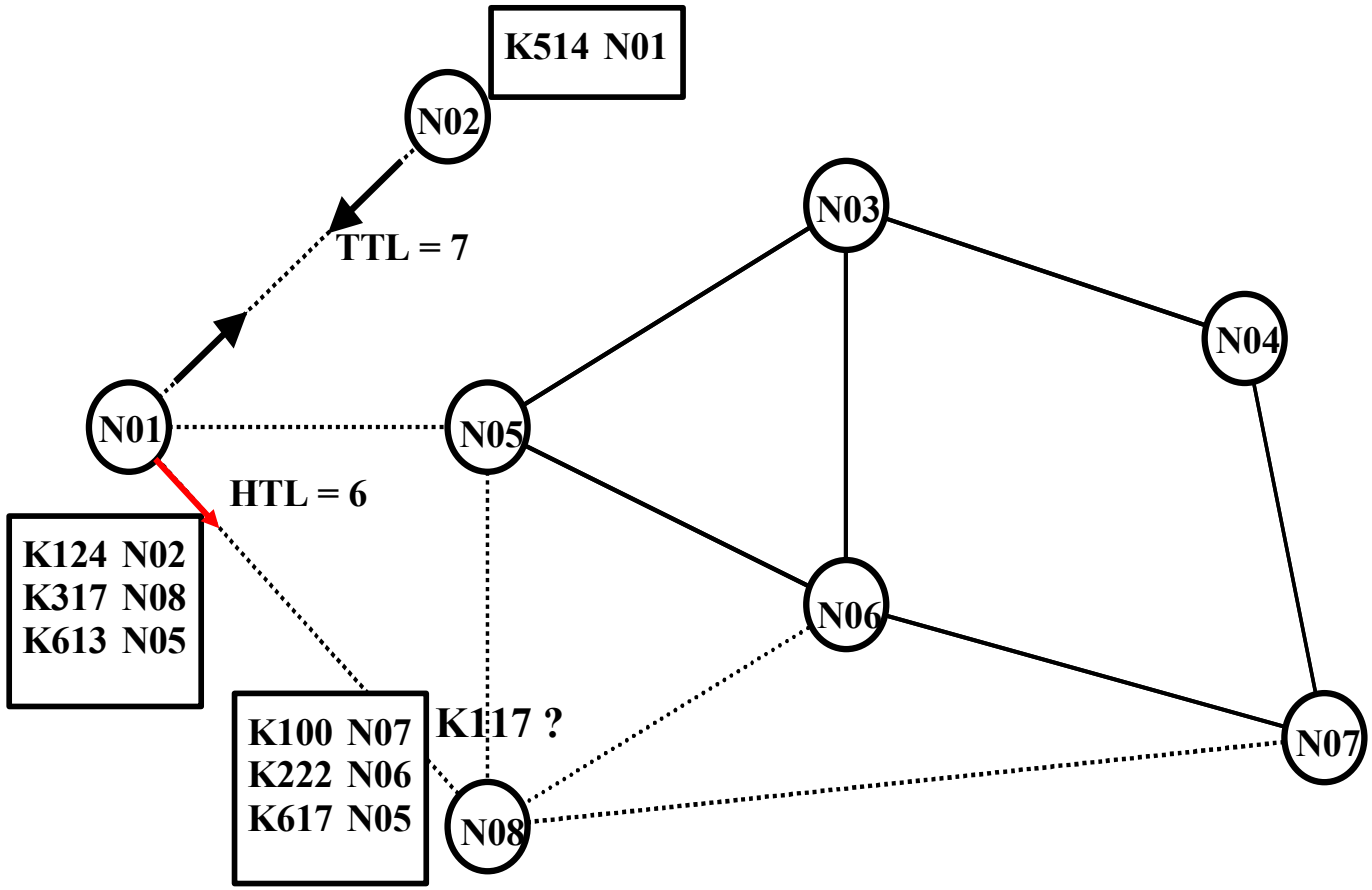
Freenet Routing



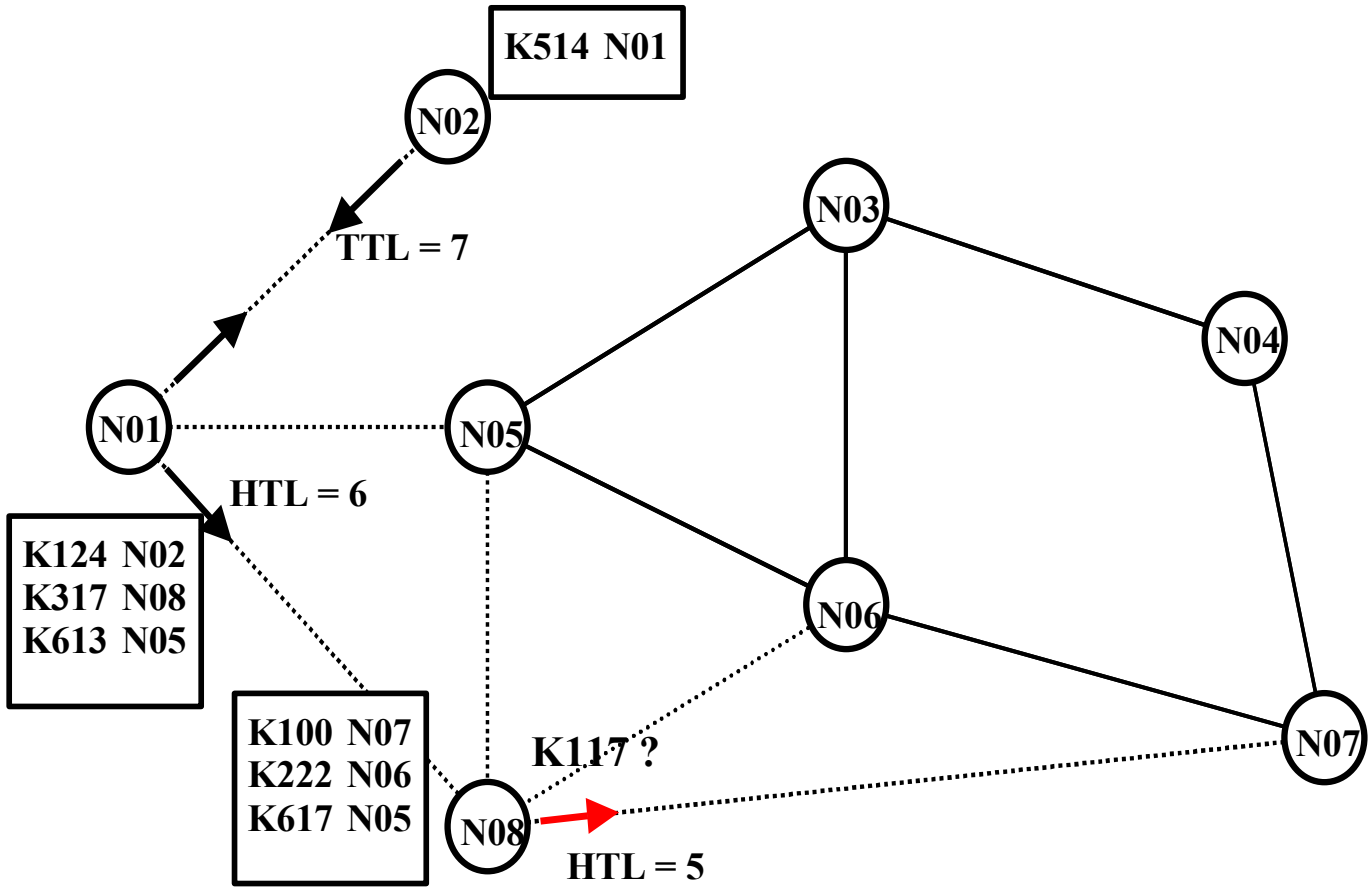
ROUTING FREENET



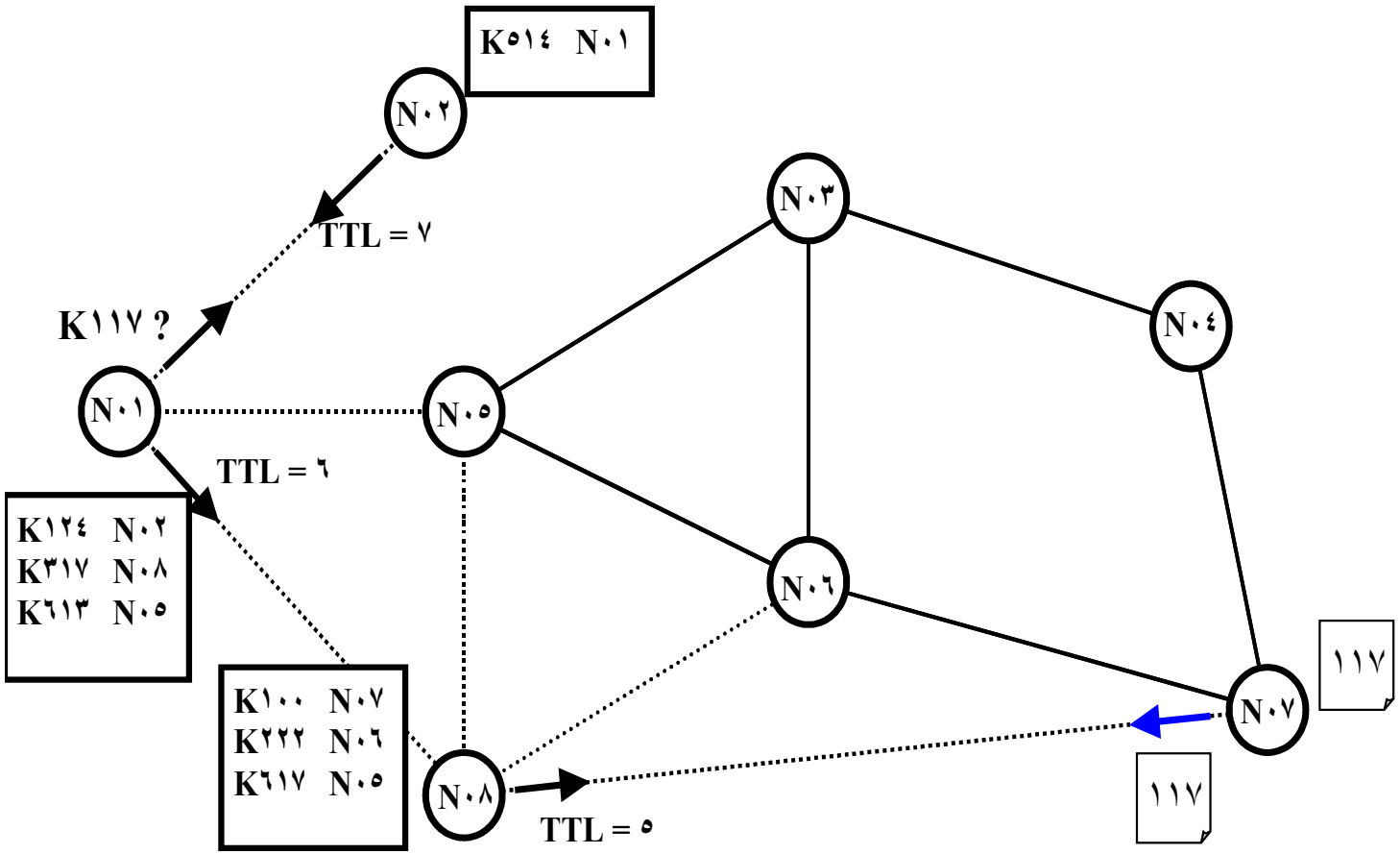
ROUTING FREENET



Freenet Routing

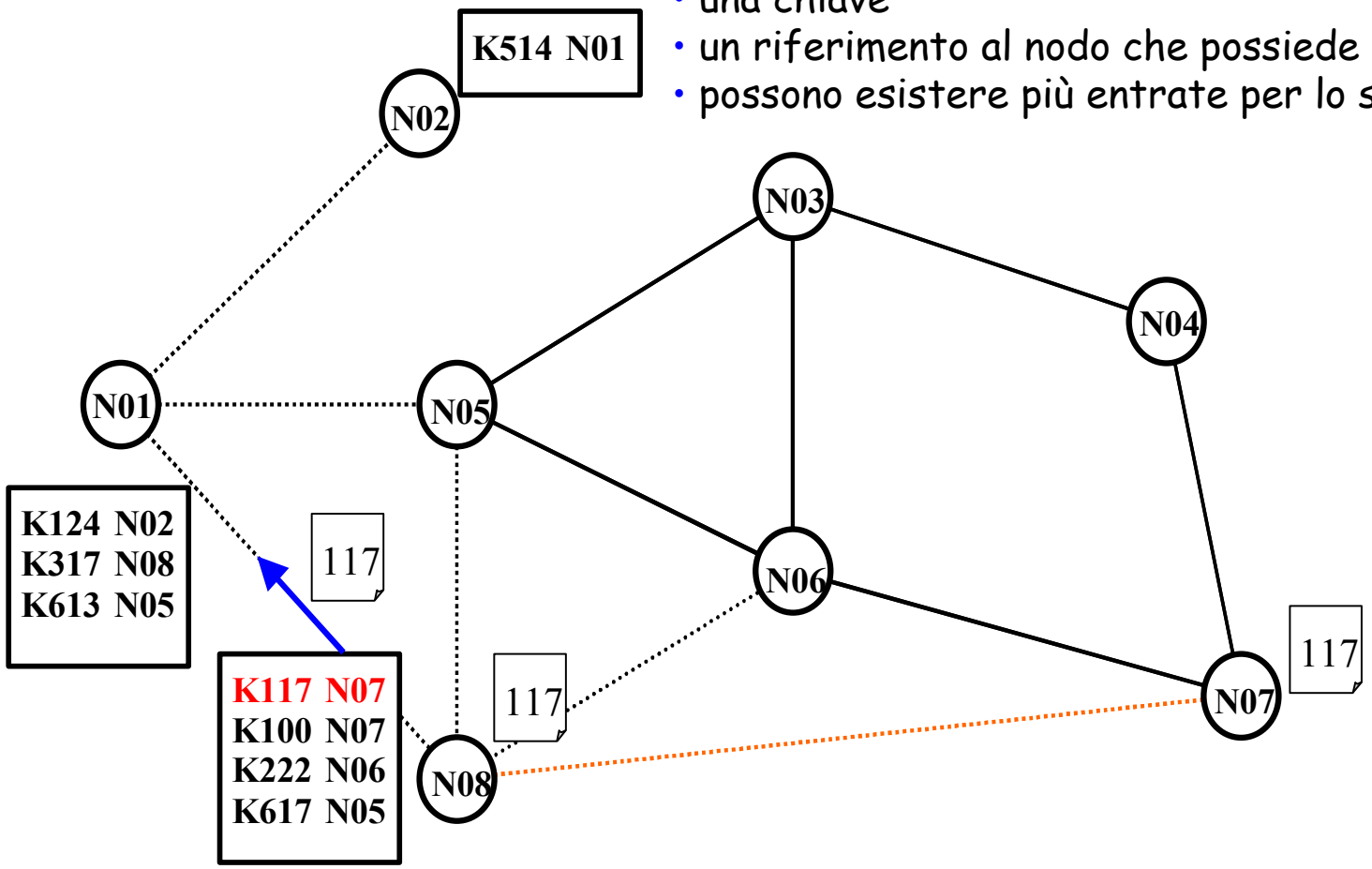


ROUTING FREENET



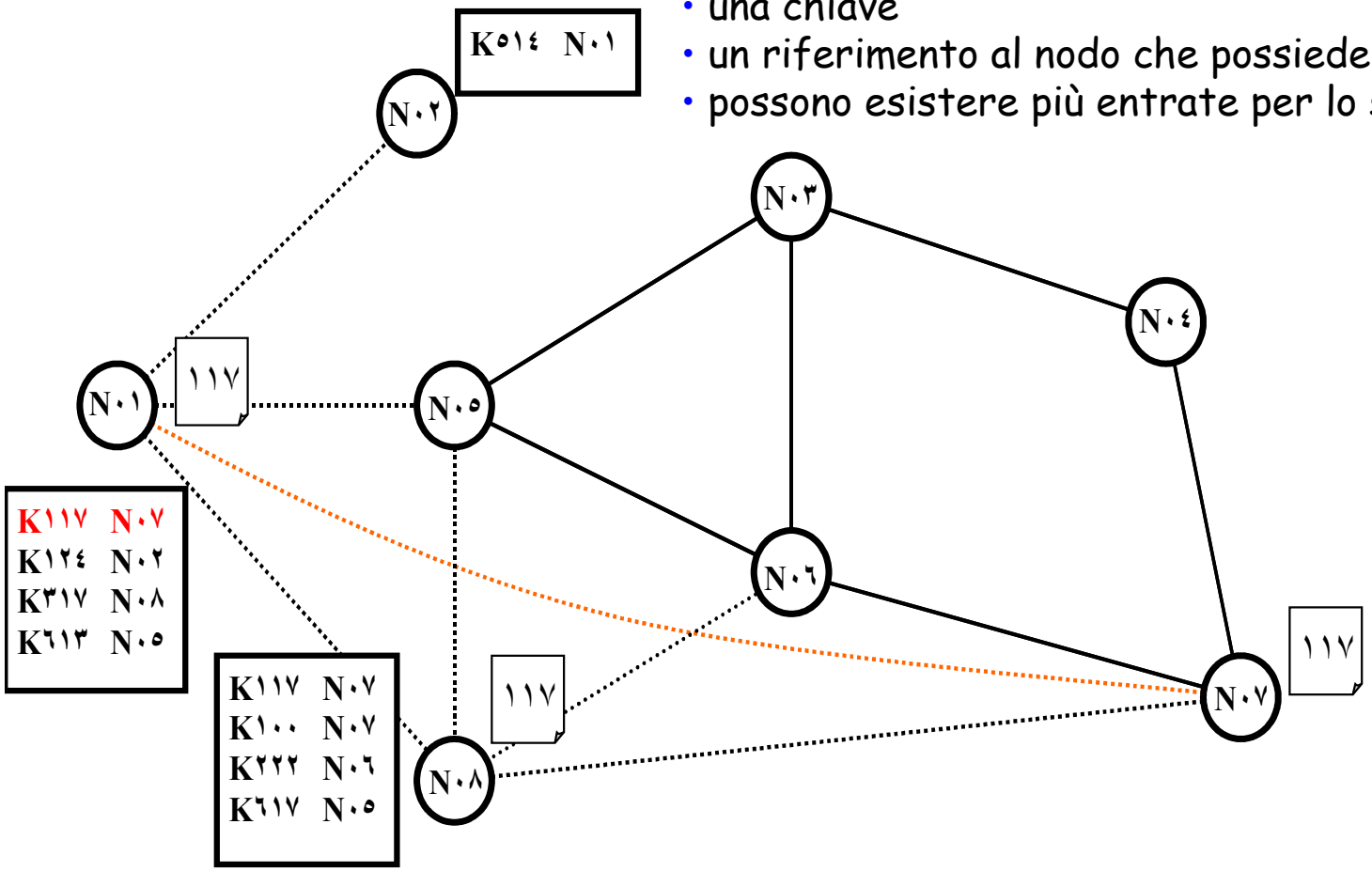
ROUTING FREENET

Routing Table: ogni entrata contiene
- una chiave
- un riferimento al nodo che possiede la chiave
- possono esistere più entrate per lo stesso nodo



ROUTING FREENET

Routing Table: ogni entrata contiene
- una chiave
- un riferimento al nodo che possiede la chiave
- possono esistere più entrate per lo stesso nodo



FREENET: REQUEST FAILED MESSAGES

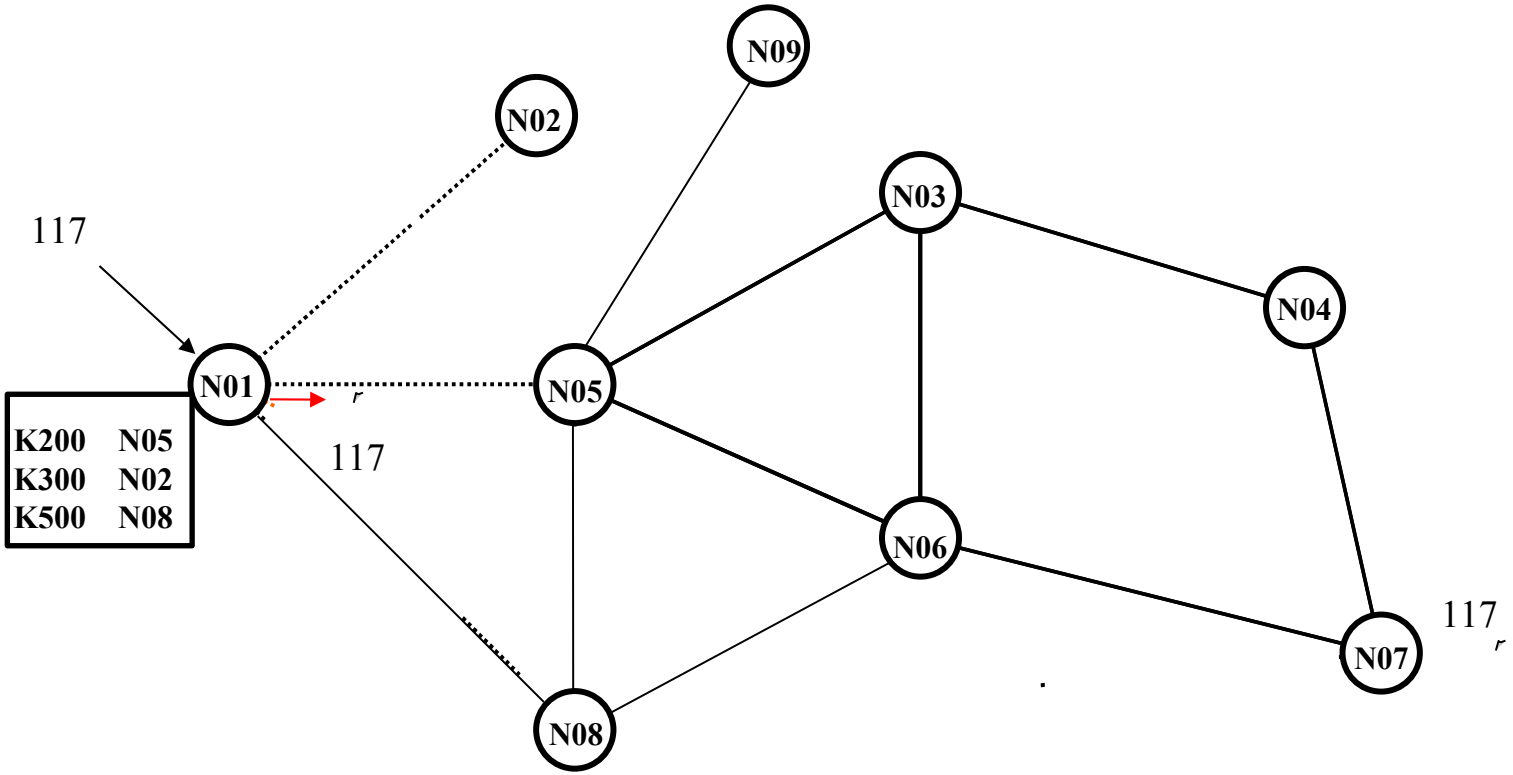
Request failed messages: vengono inviati quando

- un nodo non possiede la chiave K e
  - non possiede riferimenti a nodi vicini

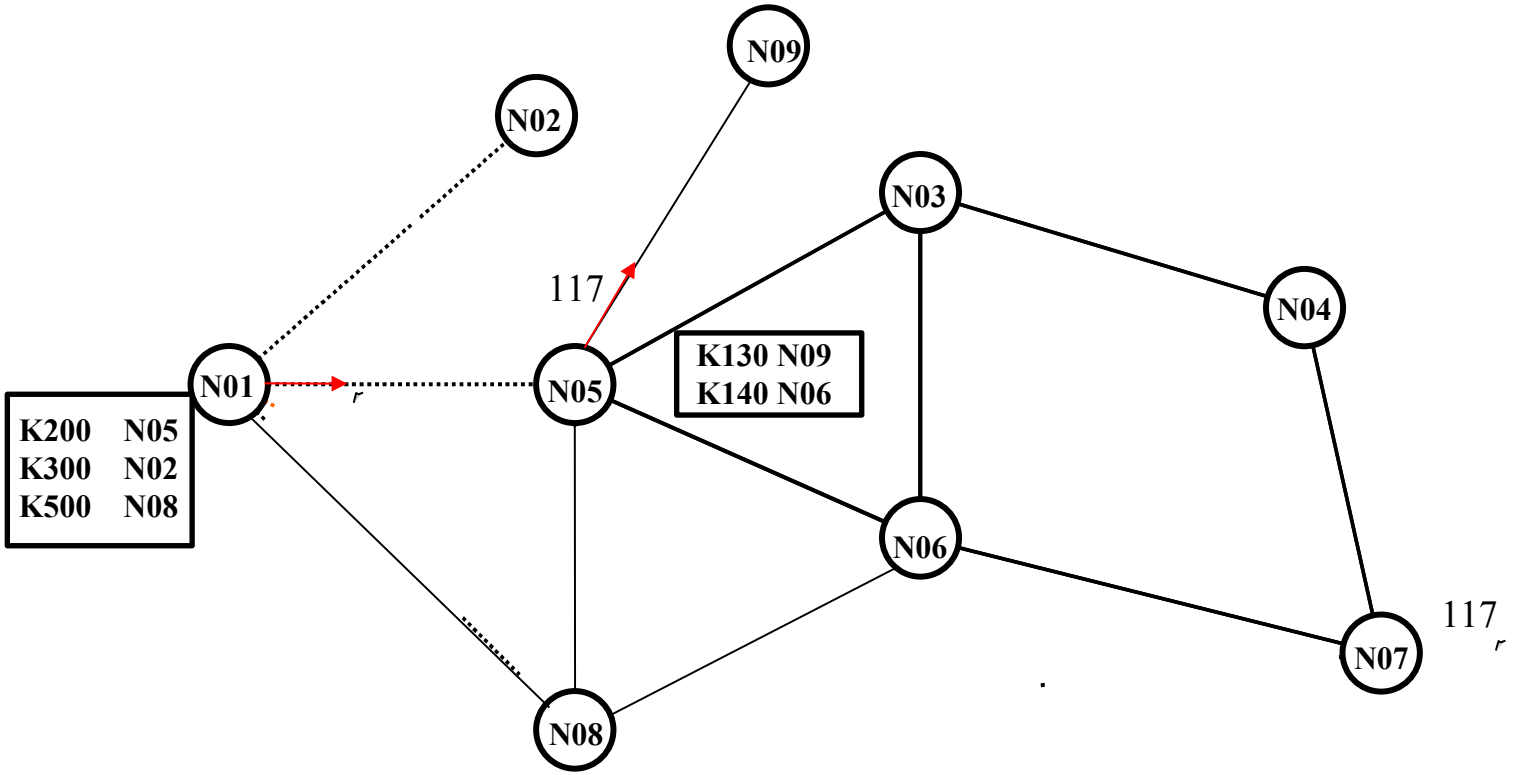
    oppure
  - i vicini contattati non posseggono K
- un nodo individua un ciclo nel routing: il nodo ha già inoltrato in precedenza k
- Quando un nodo riceve un messaggio di request failed, invia K al vicino

che possiede la chiave numericamente più vicina a K, scelta tra le chiavi rimanenti

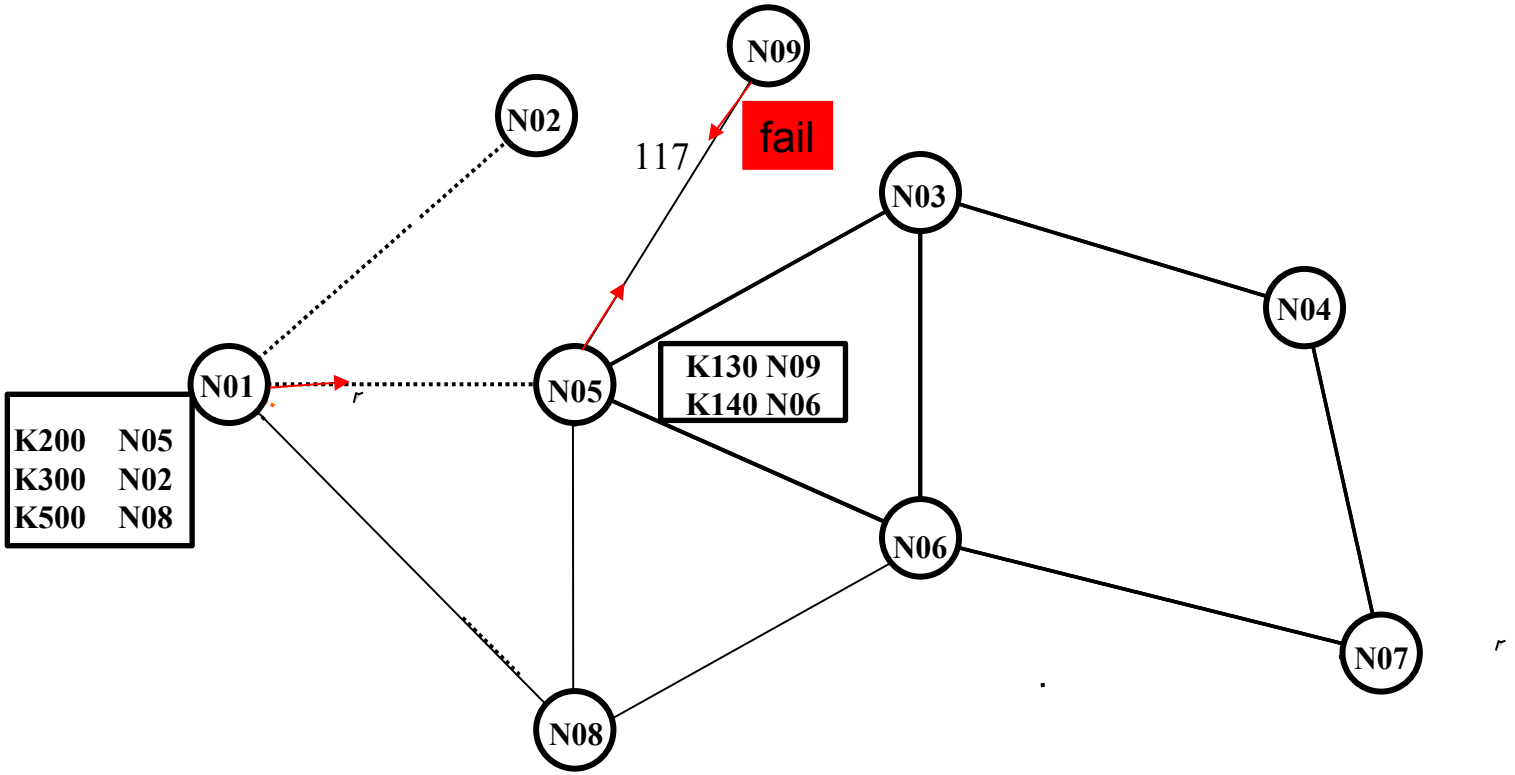
Freenet Routing



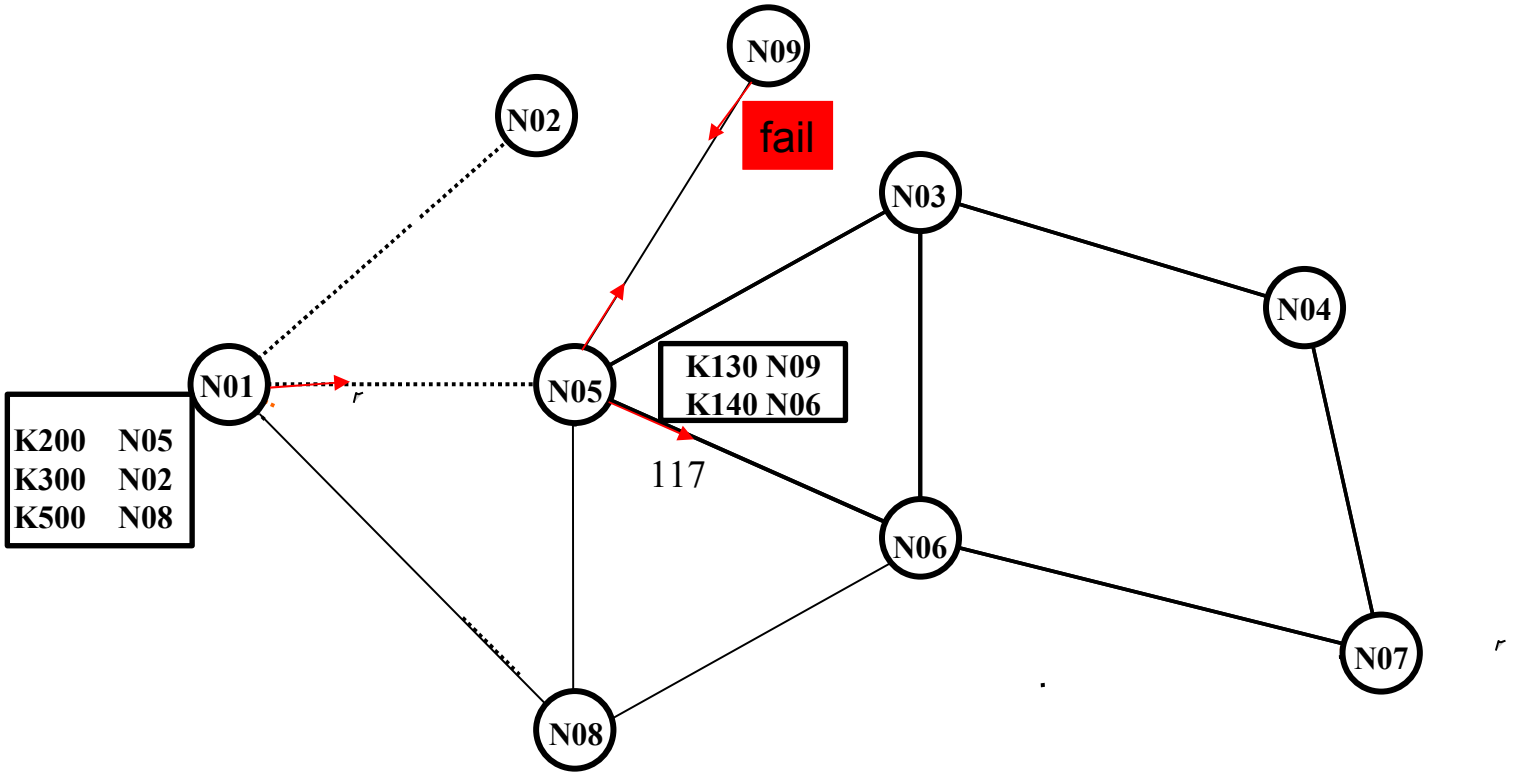
Freenet Routing



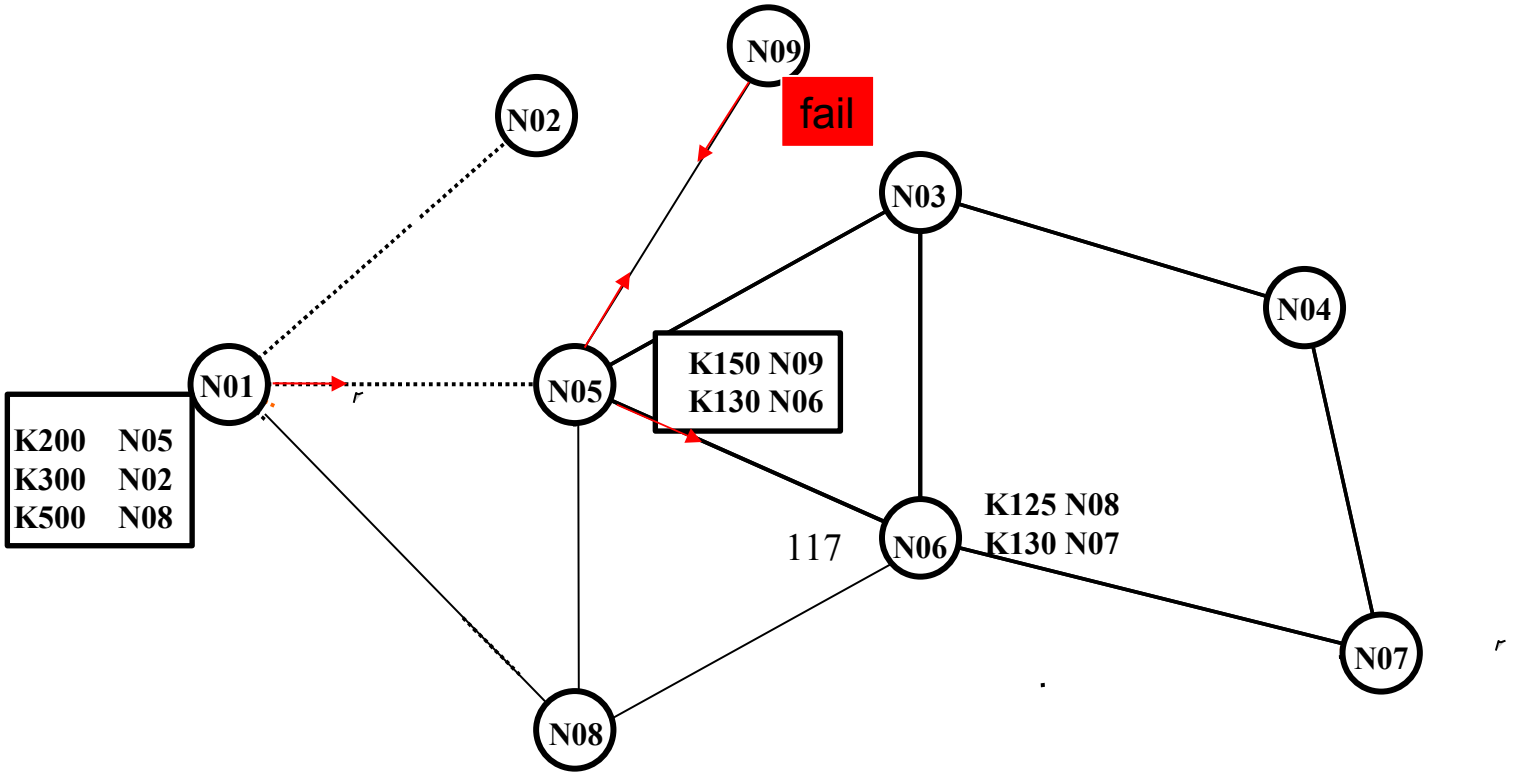
Freenet Routing



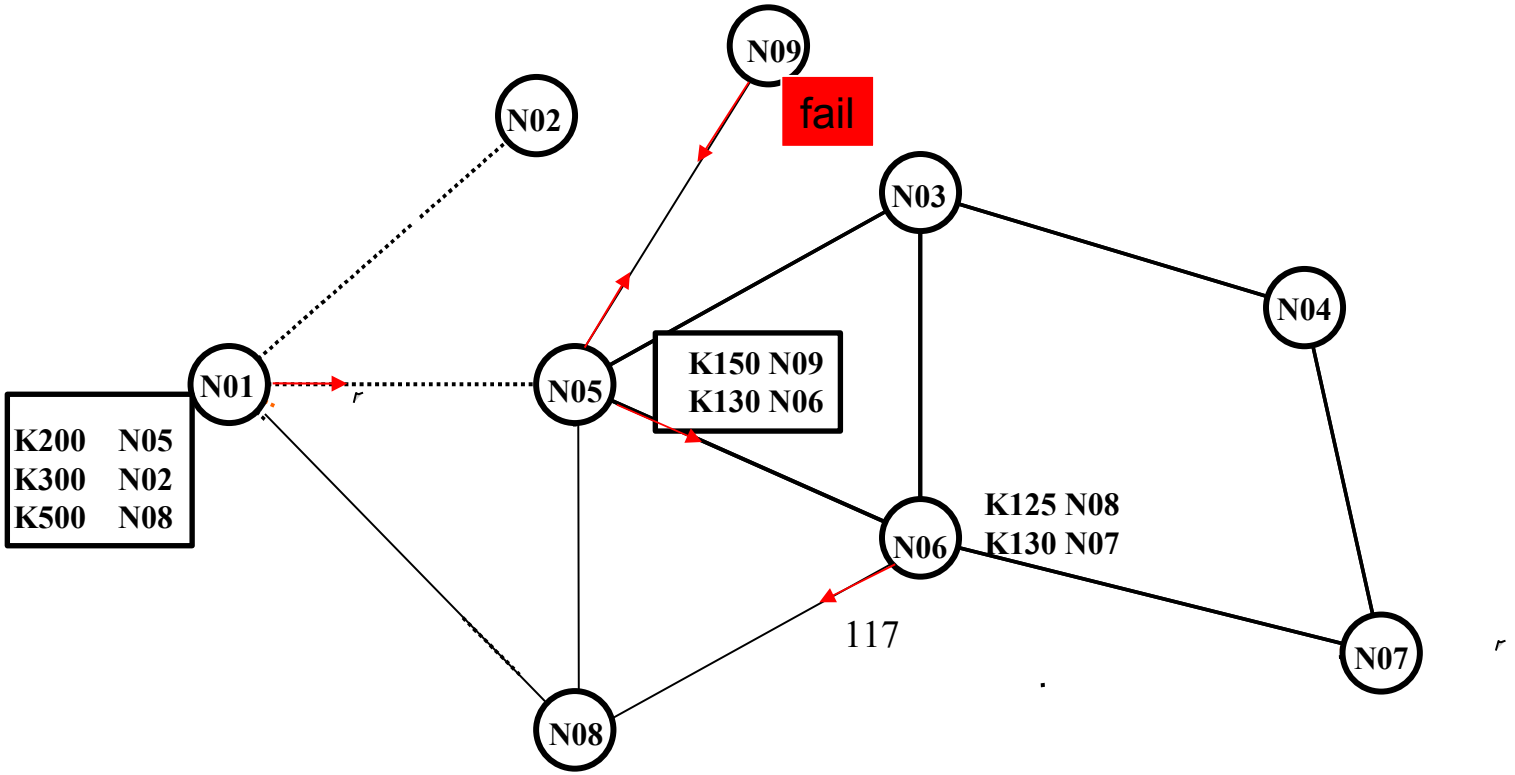
Freenet Routing



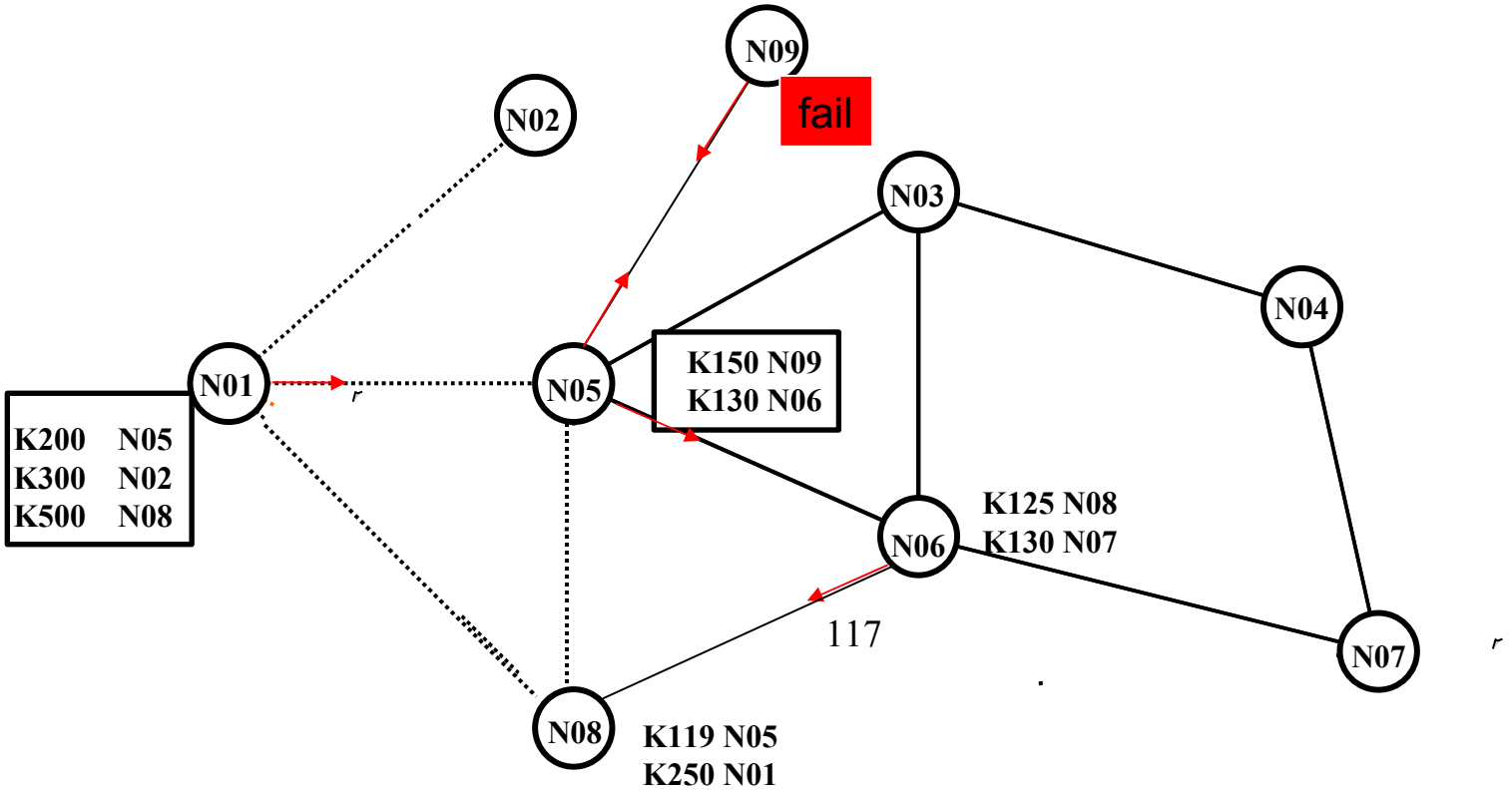
Freenet Routing



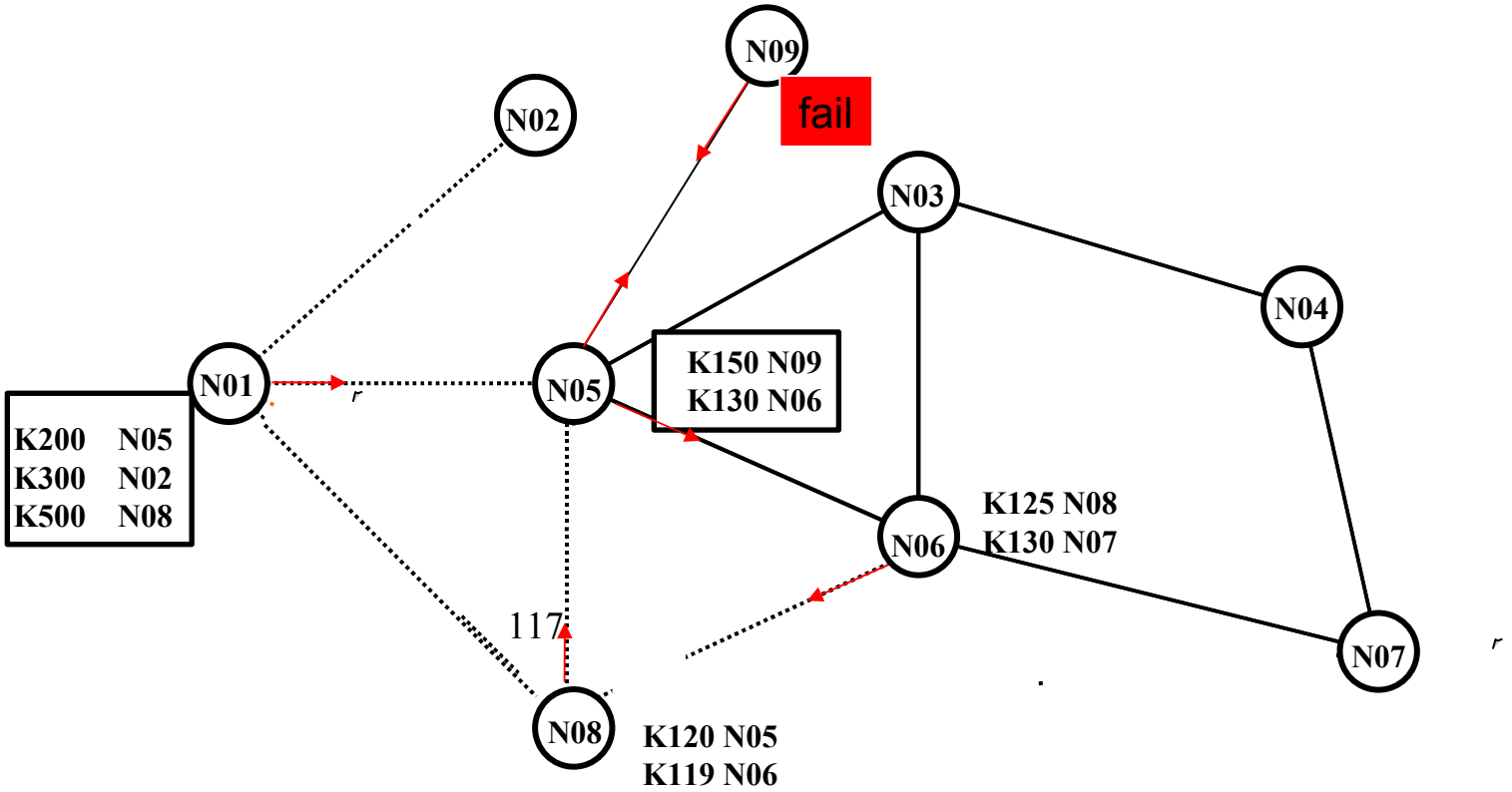
Freenet Routing



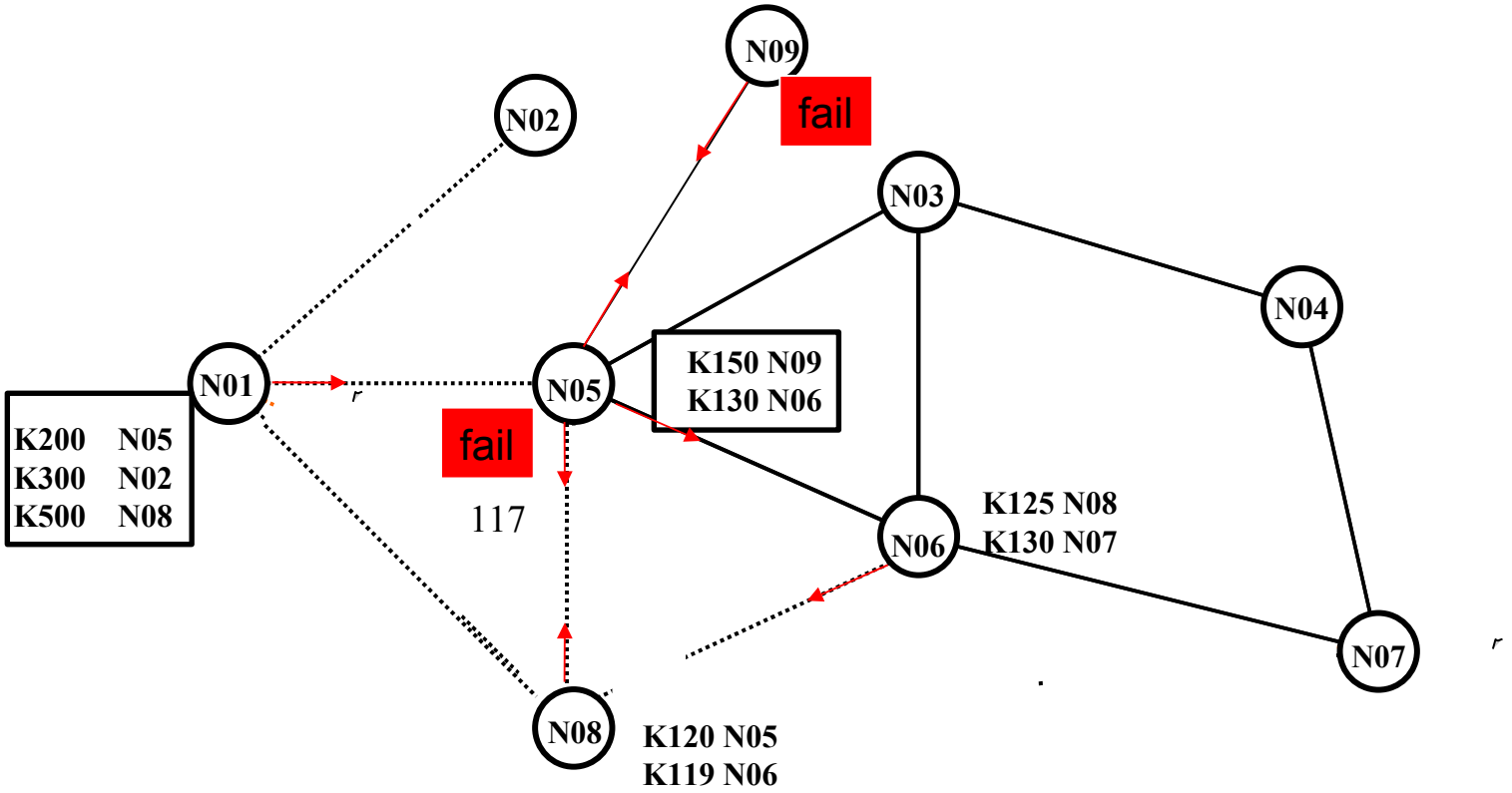
Freenet Routing



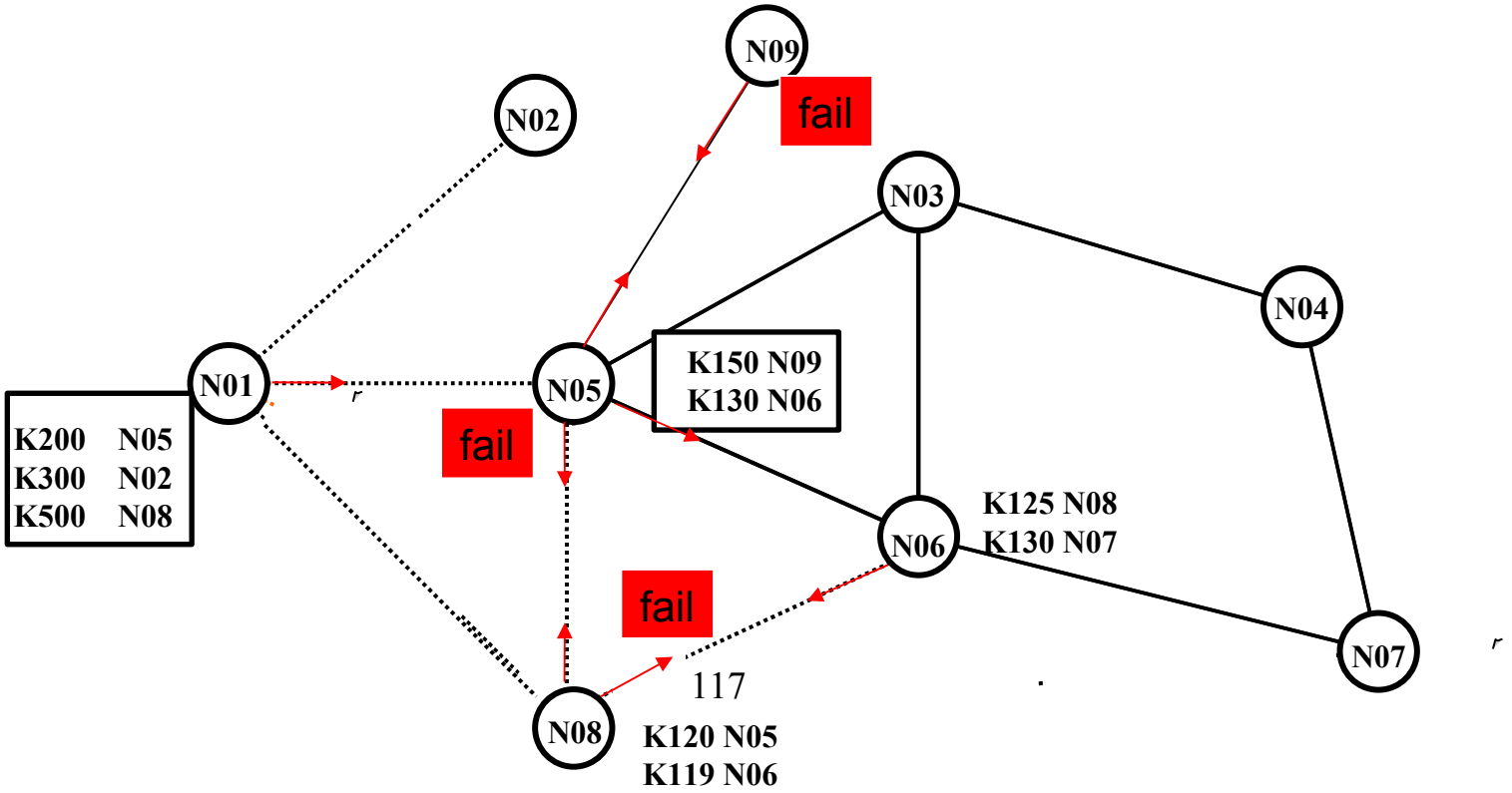
Freenet Routing



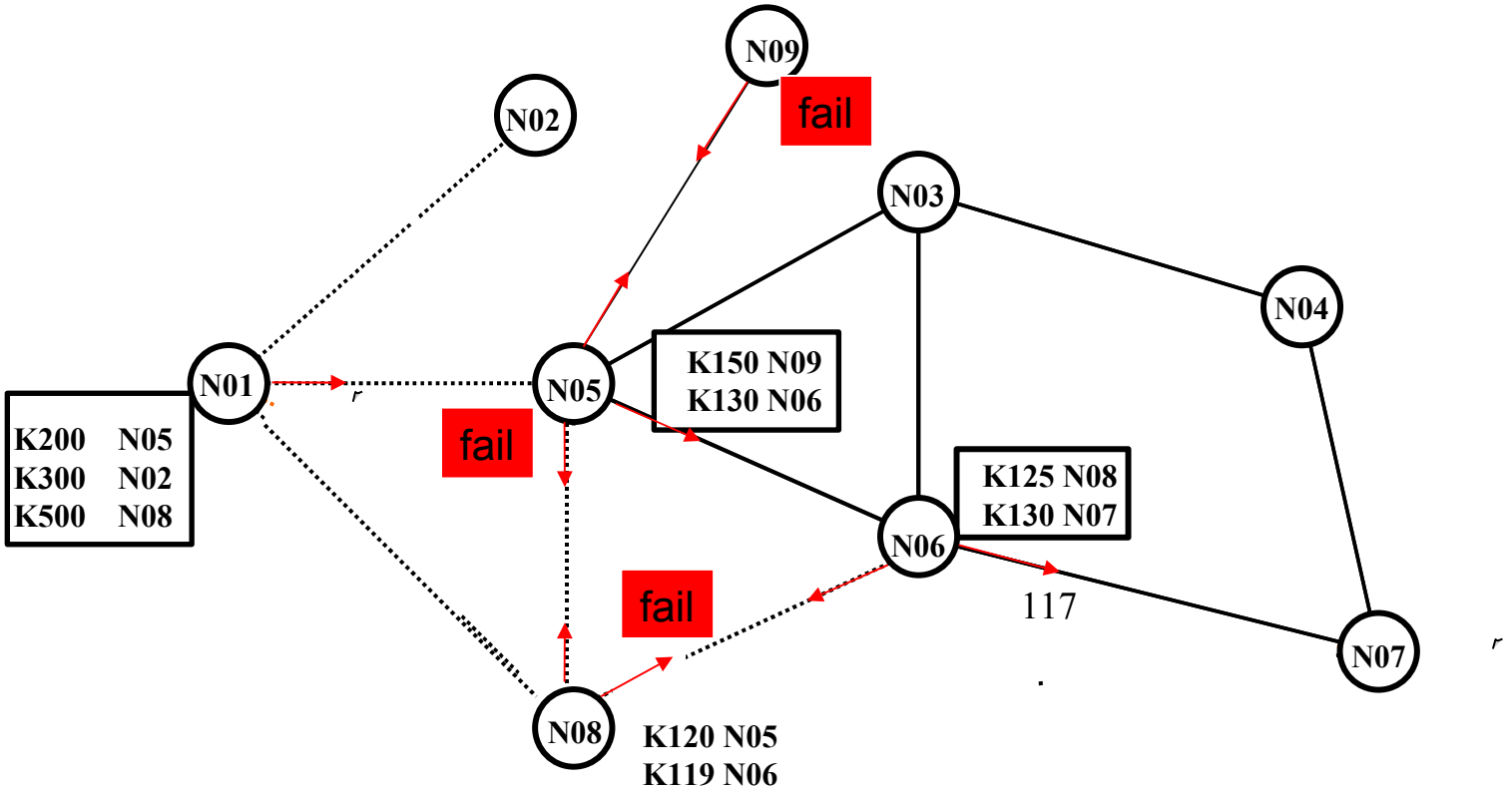
Freenet Routing



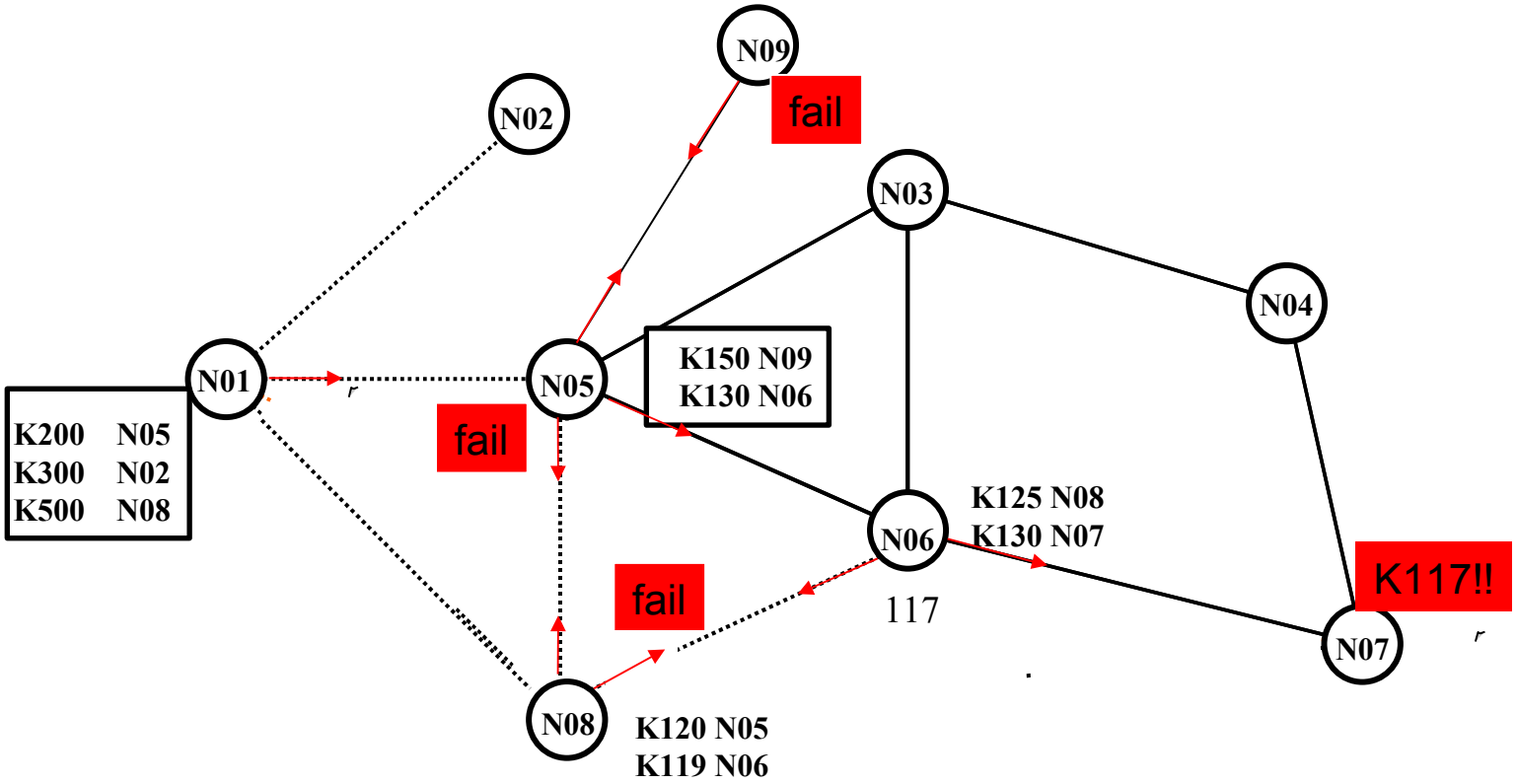
Freenet Routing



Freenet Routing



Freenet Routing



FREENET: PUBBLICAZIONE DI DATI CONDIVISI

Per pubblicare un file F su una rete Freenet, un nodo:

- assegna un una chiave binaria ad F (utilizzando una delle procedure analizzate in precedenza)
- controlla se la chiave generata è già esistente sul nodo locale. In tal caso restituisce il file corrispondente, in modo da segnalare all'utente il verificarsi di una collisione
- se la chiave non esiste
  - analizza le chiavi memorizzate nella propria tabella di routing
  - invia un messaggio insert(K,HTL) al nodo che possiede la chiave il cui valore numerico si avvicina maggiormente a K.
  - il valore del HTL (Hops to Live) indica il numero di copie di D che devono essere memorizzate nella rete

FREENET:PUBBLICAZIONE DEI DATI CONDIVISI

quando un nodo riceve un messaggio insert(K,HTL)

- controlla se possiede K, in questo caso restituisce un messaggio di collisione. Il messaggio viene propagato indietro fino al nodo sorgente.

- in caso contrario decrementa HTL
  - se il valore di HTL è diverso da 0, inoltra K al vicino che possiede la chiave numericamente più vicina a K.
  - se il valore di HTL è uguale a 0, invia indietro lungo il percorso individuato dal messaggio di insert un messaggio "all clear". Questo messaggio indica che non si sono verificate collisioni lungo il cammino percorso.

- se non si è verificata collisione, il messaggio viene memorizzato in ogni nodo lungo il percorso individuato dal messaggio di insert(K,HTL)

FREENET: AUTO ORGANIZZAZIONE

Auto organizzazione: la rete si auto-adatta durante la vita del sistema, migliorando così la qualità del routing, in seguito a:

- Clusterizzazione delle informazioni contenute nelle tabelle di routing: ogni nodo "si specializza" naturalmente nella localizzazione di chiavi simili

  Esempio:
  - il peer P è associato nelle tabelle di routing dei vicini alla chiave K.
  - P riceve queries per chiavi simili alla K. Di conseguenza P diviene sempre più esperto nell'individuare chiavi simili a K.

- Replicazione trasparente dei dati richiesti di frequente.

  Esempio:
  - un file memorizzato su un peer a Sydney è richiesto a Pisa. Il peer di Pisa lo memorizza nella propria cache. Eventualmente il file può essere replicato sui peers incontrati nel cammino da Sydney a Pisa

FREENET:AUTO ORGANIZZAZIONE

Autoorganizzazione delle tabelle di routing

- Le tabelle di routing vengono modificate durante la vita del sistema
- Quando un nodo N riceve la risposta ad una query e rileva il nodo sorgente S del file richiesto, N crea un collegamento diretto verso S, bypassando i nodi intermedi
- I nodi che forniscono dati, vengono inseriti nelle tabelle di routing degli altri nodi e verranno contattati più spesso
- La overlay network Freenet viene aggiornata dinamicamente in modo automatico

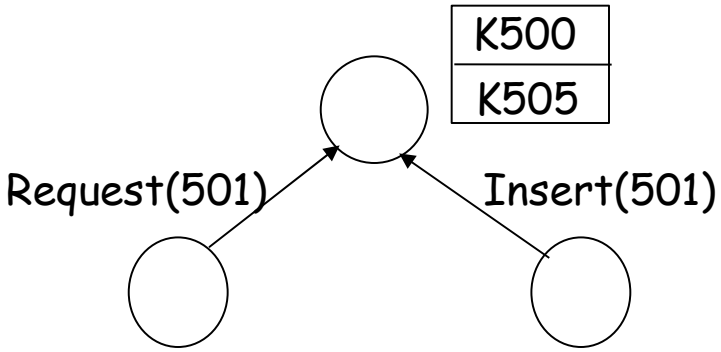
FREENET: AUTO ORGANIZZAZIONE

Clusterizzazione dei dati caratterizzati da chiavi simili nelle memorie dei peer:

- il peer P memorizza la chiave 500
- il peer PN vicino a P riceve il messaggio insert(505, htl),
- PN probabilmente invierà a P il messaggio

- Quando i cammini seguiti dalle queries e dalle richieste di memorizzazione di "si incrociano" sulla rete i cluster tendono ad aumentare di dimensione

  Esempio:



FREENET VS: DHT

Distributed Hash Tables:

- ogni chiave viene associata in modo deterministico ad uno ed un solo nodo della rete (esempio: il nodo successore sull'anello, per Chord). Questo consente di limitare la complessità della ricerca.
- la replicazione delle chiavi è introdotta solo per aumentare l'affidabilità del sistema.
- non producono falsi negativi, garantiscono limiti di complessità per algoritmi di ricerca

Freenet:

- ogni chiave viene allocata su un insieme di nodi. L'algoritmo di routing tende a clusterizzare chiavi simili sullo stesso nodo.
- può produrre falsi negativi, complessità di ricerca limitata mediante HTL.
- obiettivo del progetto: garanzia di anonimato dei nodi più che efficienza di algoritmi di ricerca

FREENET: GESTIONE DELLA MEMORIA

- la memoria viene gestita dai peer secondo un approccio LRU (Least Recently Used)
- i files utilizzati meno di frequente vengono eliminati per primi per far spazio a nuove memorizzazioni
- le entrate delle tabelle di routing vengono lasciate inalterate, in modo da poter recuperare in seguito i files cancellati
- se tutti i nodi decidono di eliminare un file il file non risulta più disponibile nel sistema
- In questo modo i documenti 'datati' vengono spontaneamente espulsi dal sistema
- I data store dei nodi non sono delle memorie cache, poichè non esiste alcuna memoria 'permanente' in cui si garantisce di mantenere una copia del file

FREENET: GARANTIRE L'ANONIMATO

- Obiettivi; garantire l'anonimato di chi pubblica informazioni e di chi le richiede

- I nodi non sono in grado di rilevare il ruolo nella rete dei vicini con cui scambiano i messaggi
  - Esempio: il vicino è il nodo che ha pubblicato l'informazione oppure un nodo che inoltra l'informazione per conto di un altro nodo?

- Tecniche utilizzate: modifica dell'identificatore del nodo sorgente di un messaggio.
  - N riceve un messaggio che indica che il nodo P possiede un certo file F.
  - N si sostituisce a P, come sorgente del file. N memorizza comunque l'identità del vero possessore di F.
  - Le successive richieste ricevute da N per F, vengono propagate a P.

FREENET: GARANTIRE L'ANONIMATO

- Un 'rumore di fondo' probabilistico' viene inserito in tutte i meccanismi di routing per evitare che un'eventuale registrazione del traffico possa far risalire al nodo che ha effettuato la richiesta o l'inserimento dei dati

- Esempi: variazione della sorgente, variazione del HTL,....ù

- Obiettivo: permettere agli utenti la ripudiabilità di un'eventuale attribuzione di responsabilità del contenuto del suo data store

FREENET: INSERIMENTO DI NUOVI NODI

Quando un nuovo peer P si inserisce su una rete Freenet esistente:

- individua un peer PB sulla rete mediante una procedura di bootstrap
- inserisce PB nella propria tabella di routing, associandogli una chiave
- la chiave associata a PB può essere generata in modo casuale, il routing è in grado successivamente di adattare la tabella di routing di P, migliorando l'attendibilità dell'informazione contenuta
- mediante l'inoltro di queries sulla rete, P scopre nuovi vicini sulla rete. La tabella di routing di P viene automaticamente popolata, in modo dinamico, durante la permanenza di P sulla rete

FREENET: INSERIMENTO DI NUOVI NODI

Il nuovo peer P che si unisce ad una rete Freenet esistente deve

- "farsi conoscere" ad altri nodi della rete. In questo modo gli altri nodi possono inviare a P queries e nuovi file da memorizzare
- per "farsi conoscere", P invia al peer di bootstrap PB un messaggio di announcement, contenente la propria identità ed un opportuno valore del HTL
- PB sceglie, in modo casuale, un vicino a cui propagare l'announcement
- la propagazione continua, per HTL passi.
- …. continua pagina successiva

FREENET:INSERIMENTO DI NUOVI NODI

- i nodi che ricevono l'announcement devono decidere di comune accordo quali chiavi assegnare al novo nodo
- le chiavi da assegnare a P possono essere definite in modo casuale, ma tutti i nodi devono concordare sui valori scelti
- i nodi eseguono un algoritmo distribuito che consente di concordare sul valore scelto per la chiave
- Cryptographic protocol: consente di definire un valore random in modo distribuito e di garantire che nessun nodo "malizioso" menta sul valore deciso collettivamente.